TDB-ACC-NO: NN9211102
DISCLOSURE TITLE: Method of Applying, Monitoring and Recording Multiple Test Criteria.

PUBLICATION-DATA: IBM Technical Disclosure Bulletin, November 1992, US

VOLUME NUMBER: 35

ISSUE NUMBER: 6

PAGE NUMBER: 102 - 114

PUBLICATION-DATE: November 1, 1992 (19921101)

CROSS REFERENCE: 0018-8689-35-6-102

DISCLOSURE TEXT:

- Described is a method of applying, monitoring and recording multiple test criteria for the efficient simultaneous characterization of performance envelope data for multiple units under test. - The multiple test criteria method provides an optimized approach for estimating multiple boundary conditions along multiple independent unidirectional vectors where each test is applied empirically and simultaneously to all units under test. Through the use of this optimized testing method, stress data can be obtained quickly, thereby enabling the user to evaluate design capabilities. - In prior art, manual and/or exhaustive testing has been used to determine the operating space or the performance envelope for the unit under test. This envelope was determined by empirically applying test criteria to the unit under test while various stress parameters were applied. Mapping the pass or fail response of the unit under test was then performed in order to obtain an estimate of the pass/fail boundary of the product. Typically, this procedure was performed manually on one unit at a time so as to apply one or more test criteria sequentially. As automation became evident, the manual one at a time approach became obsolete. Software was developed which exhaustively maps the response of each unit under test at each stress condition. However, this approach was considered inefficient since the brute-force algorithm led to many meaningless applications of the test and was time consuming. Also, this approach lacked the means to obtain separate data for different test criteria. For example, if separate performance envelopes were being obtained for a product running test criteria A and results were also required for test criteria B, the entire test had to be run separately for each of these test criteria. - The implementation of the multiple test criteria method consists of the disclosed algorithms used to control automated test equipment. The algorithms make use of the stress vector concept where the vector involves a combination of stresses. A vector may be thought of as a stress parameter where the controlled setting is under the control of an algorithm. The vector is incremented in predetermined steps by the algorithm, and all chosen criteria are applied until all units under test fail their criteria. - The following list outlines the features provided in the multiple test criteria method: - Selectable Number of Pass/Fail Retries - The user is able to select any number of pass or fail events which must occur in a row in order for the result to be considered

valid. This allows the filtering of intermittent or spurious results. - - Selectable Combination of Search Types - The user may select one of several search algorithms which include linear searches, binary searches and a combination of linear/binary and binary/linear searches. The combination searches use one search algorithm for the gross portion of the test and another for narrowing down the pass/fail boundary, thereby enabling the boundary to be located with greater precision. The relative effectiveness of the various search types depends on many factors, including the number of samples tested, the number of test criteria and the expected consistency of the results. Providing flexibility in the search types allows the user to select the most efficient search for the application. - - Return to Nominal for Detection of Multiple Failure Points - The algorithm allows for a return to normal situation in which the stress factors are optionally returned to nominal values until the case of interest is ready to run. This allows the determination of many boundary points beyond the initial failing point where a simple algorithm would not allow the test to continue. For example, it may be desirable to complete a power on self test (POST) prior to the execution of diagnostic test cases under the control of an operating system. - - Simultaneous Characterization - Because the results for all test criteria are determined simultaneously, the results are not affected by external factors, such as time of day, variances in stress equipment, etc. Therefore, results can be compared with confidence that any differences noted are significant. - - Expandability - The algorithms can be applied to any number of samples with any number of test criteria in use. This improves the efficiency as the complexity of the test scenario increases. - - Advanced Test Data Storage, Updating and Output Features - Because actual measured data is obtained in real time rather than a pass or fail indication for predetermined stress points, the output flexibility is increased. This includes the following: - Detection and Differentiation of Multiple Failure Modes - Because both reported and hangup failures are detected and tracked separately, a better understanding of the failure is attained. - - Separate Data for Last Pass and First Fail Data - As the test proceeds, the worst-case pass data and the best-case fail data for each test criteria will be constantly updated. This allows retrieval of data from interrupt or incomplete tests and for graphical display of tests in progress. - - Multiple Output Formats - The final output data can be made available in several forms. Among these are the summary file form which contains the data in a form easily interpreted by a human operator and the envelope file form which contains the data in a form which may be input directly to a spreadsheet or graphing program. - The multiple test criteria method is considered applicable in other areas, such as: a) iterative mathematical approaches in which the operation is performed repeatedly until a certain condition is met; b) in data searches in which a segment of data is analyzed for the presence of one or more specific strings or values; and c) in automated mechanical tests such as hardness, breaking stress or tolerance testing. Additionally, the data storage format approach could be applicable in any case where many variables must be updated to reflect the state of many objects. The algorithm is equally useful for testing products at any level of integration. The flowcharts indicate a unit under test reset by the term POR for power on reset. For different products, another type of reset might be used. The term hangup refers to the detection of the condition in which the unit under test has not indicated a failure or halted during the running of a test case, but in which the unit under test has not reported or responded within the expected amount of time. This is useful in the case where a product hangs or locks up during the test when the hang cannot be attributed to the execution of a diagnostic test. - The flow charts (Figs. 1 - 10) show the various steps required by the multiple test criteria method during the processing of the algorithms. Fig. 1 shows a flow chart of the steps required to run the high-level run-time algorithm. The remaining flow charts, Figs. 2 to 10, show the steps to run the algorithms of each main sub-process. The best way to follow the flow charts is to visualize several of the units under test. For example, to determine the limits

to which the system can be stressed before failures occur requires the determination of the exact failure point at which each diagnostic test begins to fail on each of the units under test as the stress increases. The descriptions of the various flow charts are as follows: Fig. 1 - Runtime.Flo - This is a global process. At the completion of this process, all equipment is initialized. A complex series of events now occurs. In general, the method sets a stress condition, applies a test criterion, logs the results of the test and then determines the next stress condition to be set based on the results of the test. The algorithm allows simultaneous determination of many criteria for many systems. This process continues until all desired data is obtained, after which all equipment is shut down. Within the global process are several important sub-processes. Each of these is shown in its own flow chart (Figs. 2 to 10) and described below. The descriptions are simplified in that many additional checks and adjustments are made during the course of each process. Fig. 2 - INITVAR.FLO - In this sub-process, all data arrays are set up and initialized and the starting point is determined. Fig. 3 - INITVEC.FLO - In this sub-process, the initial stress point is programmed into the test equipment and verified. Then the units under test are started. Fig. 4 - PROCLOOP.FLO - In this looping sub-process, each unit under test is examined in sequence. One of several possible statuses is determined for each unit under test. If a system is determined to have failed a test, this information is logged. The loop continues until the status of all tests on all units under test have been determined. Fig. 5 - PROCPAR.FLO - In this sub-process, communications from the units under test are interpreted and reacted to. The units under test can be sending one of three types of messages to the control system. The first two of these are the messages that a particular test passed or failed. When either of these occurs, the data points associated with that test on that unit under test are updated. The other possible message is a request for permission to run a particular test. When this occurs, the control system consults several rules, taking into account whether the operator has selected that test, whether that test has failed previously, etc., and responds appropriately to the unit under test. This sub-process also tracts and reacts to the status unit under test, such as if each system has completed its test, then the next step is initiated. Fig. 6 - CHKDONE.FLO - This process determines whether a particular test vector has been completed for any units under test and returns its decision to the main processor. At each invocation, it determines if any unit under test can be eliminated from future tests. When all units under test have been eliminated, the vector is complete. Fig. 7 - NEXTSTEP.FLO - This sub-process resets all data arrays and prepares the test system to run the tests at the next stress point. Figs. 8 & 9 - RECOVER.FLO/RECNOM.FLO - These sub-processes occur when a unit under test has hung. They conditionally bring the stress conditions back to nominal and reset the unit under test, allowing it to recover from the hang. Fig. 10 - WAITPOST.FLO - This is a special sub-process which occurs between the time that the units under test are initialized and the time that they report that they have successfully brought themselves up. Hangups or failures during initialization are detected here. - Not shown in the flow charts are the following periodic checks, all of which halt the algorithms upon failure: - Every five seconds, read the status of all connected equipment, such as temperature chambers and power supplies. Halt upon report of abnormal status. - Every five seconds, read the current values from all connected equipment and compare each value read with the acceptable range of equipment. Halt if any parameter exceeds the acceptable equipment ranges. - Every sixty seconds, read all data logger channels and compare each value read with the range of the equipment unit connected to the channel. Halt upon detection of an out of range condition. This provides an independent check on the current stress values being applied to the units under test. - Every five seconds, read current temperatures from temperature chambers and compare each value read with the

desired value of the temperature chamber. This assumes that the temperatures have not gone out of control.

SECURITY: Use, copying and distribution of this data is subject to the restictions in the Agreement For IBM TDB Database and Related Computer Databases. Unpublished - all rights reserved under the Copyright Laws of the United States. Contains confidential commercial information of IBM exempt from FOIA disclosure per 5 U.S.C. 552(b)(4) and protected under the Trade Secrets Act, 18 U.S.C. 1905.

COPYRIGHT STATEMENT: The text of this article is Copyrighted (c) IBM Corporation 1992. All rights reserved.

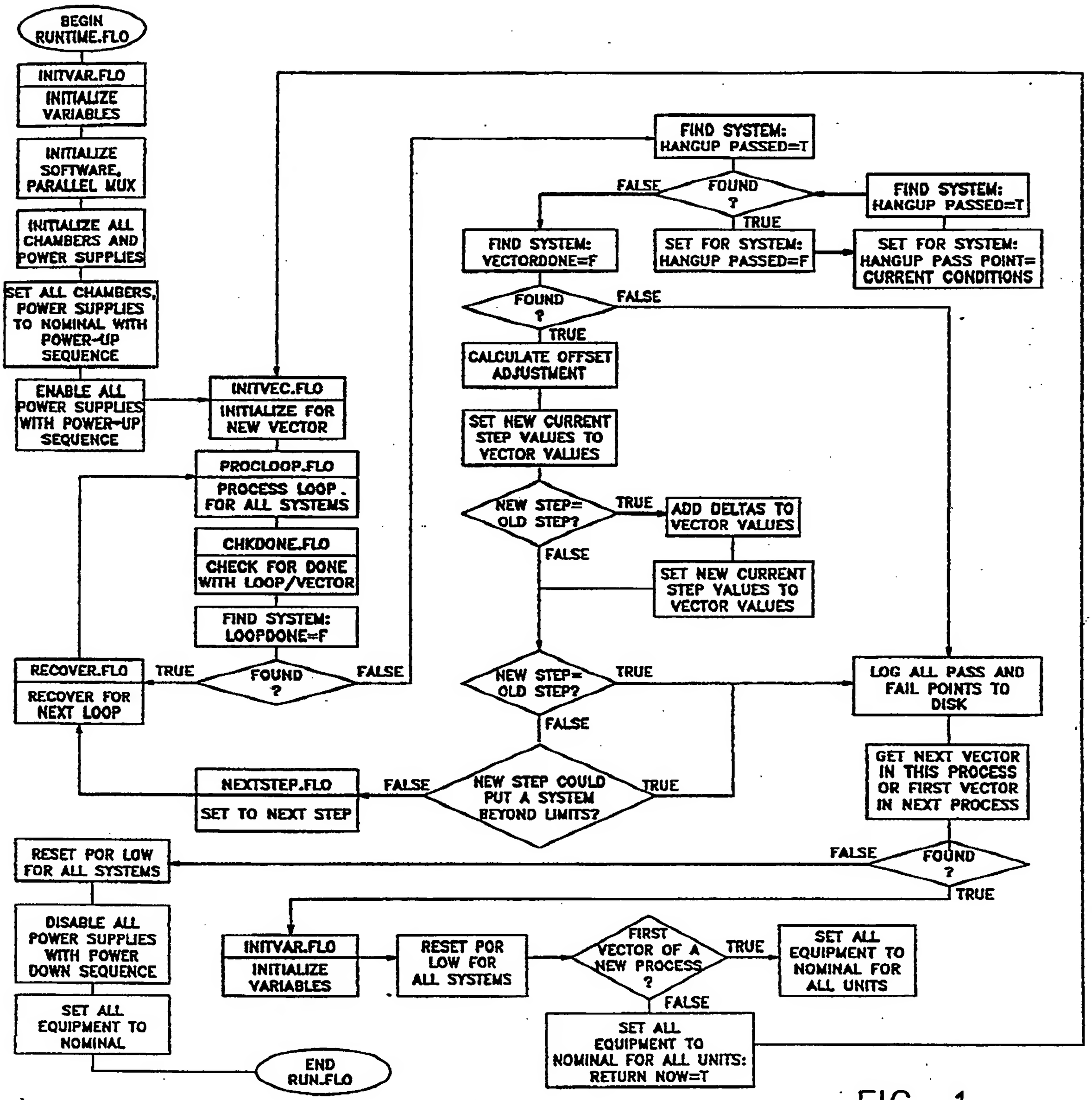
BEGIN RUNTIME.FLO
INITVAR.FLO
INITIALIZE VARIABLES
INITIALIZE SOFTWARE, PARALLEL MUX
INITIALIZE ALL CHAMBERS AND POWER SUPPLIES
SET ALL CHAMBERS, POWER SUPPLIES TO NOMINAL WITH POWER-UP SEQUENCE
ENABLE ALL POWER SUPPLIES WITH POWER-UP SEQUENCE
INITVEC.FLO
INITIALIZE FOR NEW VECTOR
PROCLOOP.FLO
PROCESS LOOP. FOR ALL SYSTEMS
CHKDONE.FLO
CHECK FOR DONE WITH LOOP/VECTOR
FIND SYSTEM: LOOPDONE=F
FOUND ?
TRUE
FALSE
RECOVER.FLO
RECOVER FOR NEXT LOOP
FIND SYSTEM: VECTORDONE=F
FOUND ?
FALSE
TRUE
CALCULATE OFFSET ADJUSTMENT
SET NEW CURRENT STEP VALUES TO VECTOR VALUES
NEW STEP= OLD STEP?
TRUE
ADD DELTAS TO VECTOR VALUES
FALSE
SET NEW CURRENT STEP VALUES TO VECTOR VALUES
NEW STEP= OLD STEP?
TRUE
FALSE
NEW STEP COULD PUT A SYSTEM BEYOND LIMITS?
FALSE
TRUE
NEXTSTEP.FLO
SET TO NEXT STEP
FIND SYSTEM: HANGUP PASSED=T
FOUND ?
FALSE
TRUE
FIND SYSTEM: HANGUP PASSED=T
SET FOR SYSTEM: HANGUP PASSED=F
SET FOR SYSTEM: HANGUP PASS POINT= CURRENT CONDITIONS
LOG ALL PASS AND FAIL POINTS TO DISK
GET NEXT VECTOR IN THIS PROCESS OR FIRST VECTOR IN NEXT PROCESS
FOUND ?
FALSE
TRUE
RESET POR LOW FOR ALL SYSTEMS
DISABLE ALL POWER SUPPLIES WITH POWER DOWN SEQUENCE
SET ALL EQUIPMENT TO NOMINAL
END RUN.FLO
INITVAR.FLO
INITIALIZE VARIABLES
RESET POR LOW FOR ALL SYSTEMS
FIRST VECTOR OF A NEW PROCESS ?
TRUE
SET ALL EQUIPMENT TO NOMINAL FOR ALL UNITS
FALSE
SET ALL EQUIPMENT TO NOMINAL FOR ALL UNITS: RETURN NOW=T

FIG. 1

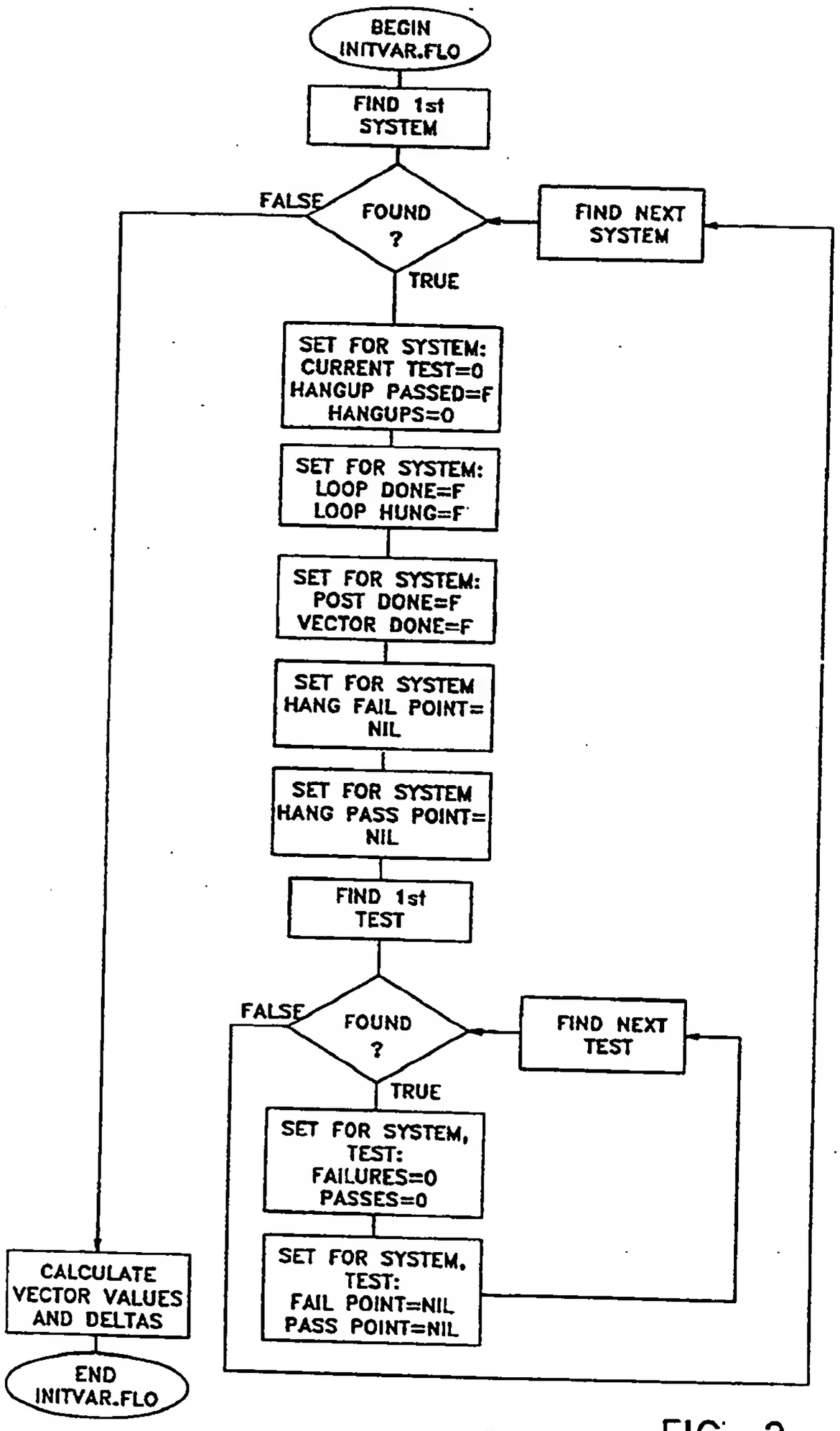
BEGIN INITVAR.FLO
FIND 1st SYSTEM
FOUND ?
FALSE
TRUE
FIND NEXT SYSTEM
SET FOR SYSTEM: CURRENT TEST=0 HANGUP PASSED=F HANGUPS=0
SET FOR SYSTEM: LOOP DONE=F LOOP HUNG=F
SET FOR SYSTEM: POST DONE=F VECTOR DONE=F
SET FOR SYSTEM HANG FAIL POINT= NIL
SET FOR SYSTEM HANG PASS POINT= NIL
FIND 1st TEST
FOUND ?
FALSE
TRUE
FIND NEXT TEST
SET FOR SYSTEM, TEST: FAILURES=0 PASSES=0
SET FOR SYSTEM, TEST: FAIL POINT=NIL PASS POINT=NIL
CALCULATE VECTOR VALUES AND DELTAS
END INITVAR.FLO

FIG. 2

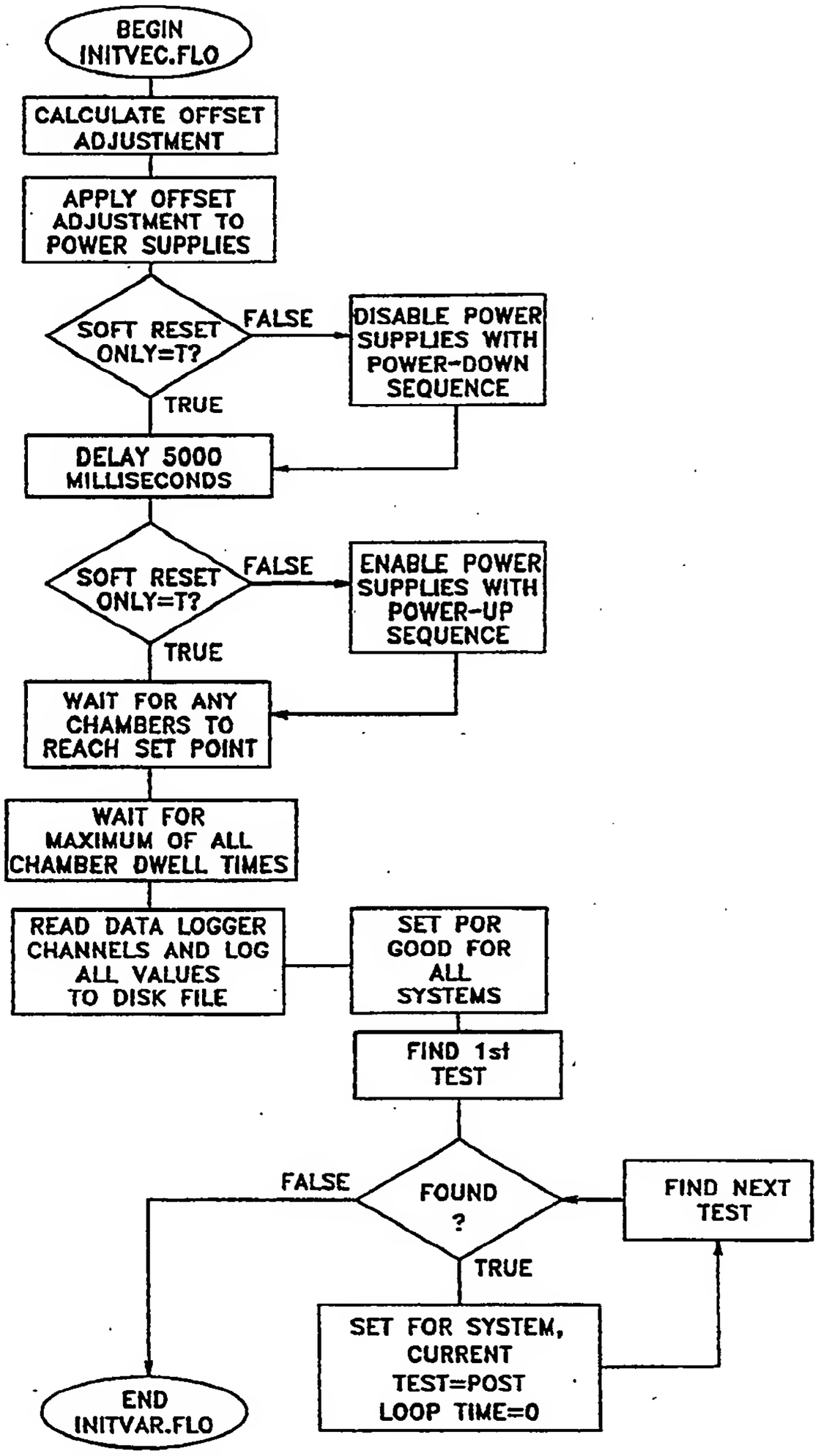
BEGIN INITVEC.FLO
CALCULATE OFFSET ADJUSTMENT
APPLY OFFSET ADJUSTMENT TO POWER SUPPLIES
SOFT RESET ONLY=T?
FALSE
TRUE
DISABLE POWER SUPPLIES WITH POWER-DOWN SEQUENCE
DELAY 5000 MILLISECONDS
SOFT RESET ONLY=T?
FALSE
TRUE
ENABLE POWER SUPPLIES WITH POWER-UP SEQUENCE
WAIT FOR ANY CHAMBERS TO REACH SET POINT
WAIT FOR MAXIMUM OF ALL CHAMBER DWELL TIMES
READ DATA LOGGER CHANNELS AND LOG ALL VALUES TO DISK FILE
SET POR GOOD FOR ALL SYSTEMS
FIND 1st TEST
FOUND ?
FALSE
TRUE
FIND NEXT TEST
SET FOR SYSTEM, CURRENT TEST=POST LOOP TIME=0
END INITVAR.FLO

FIG. 3

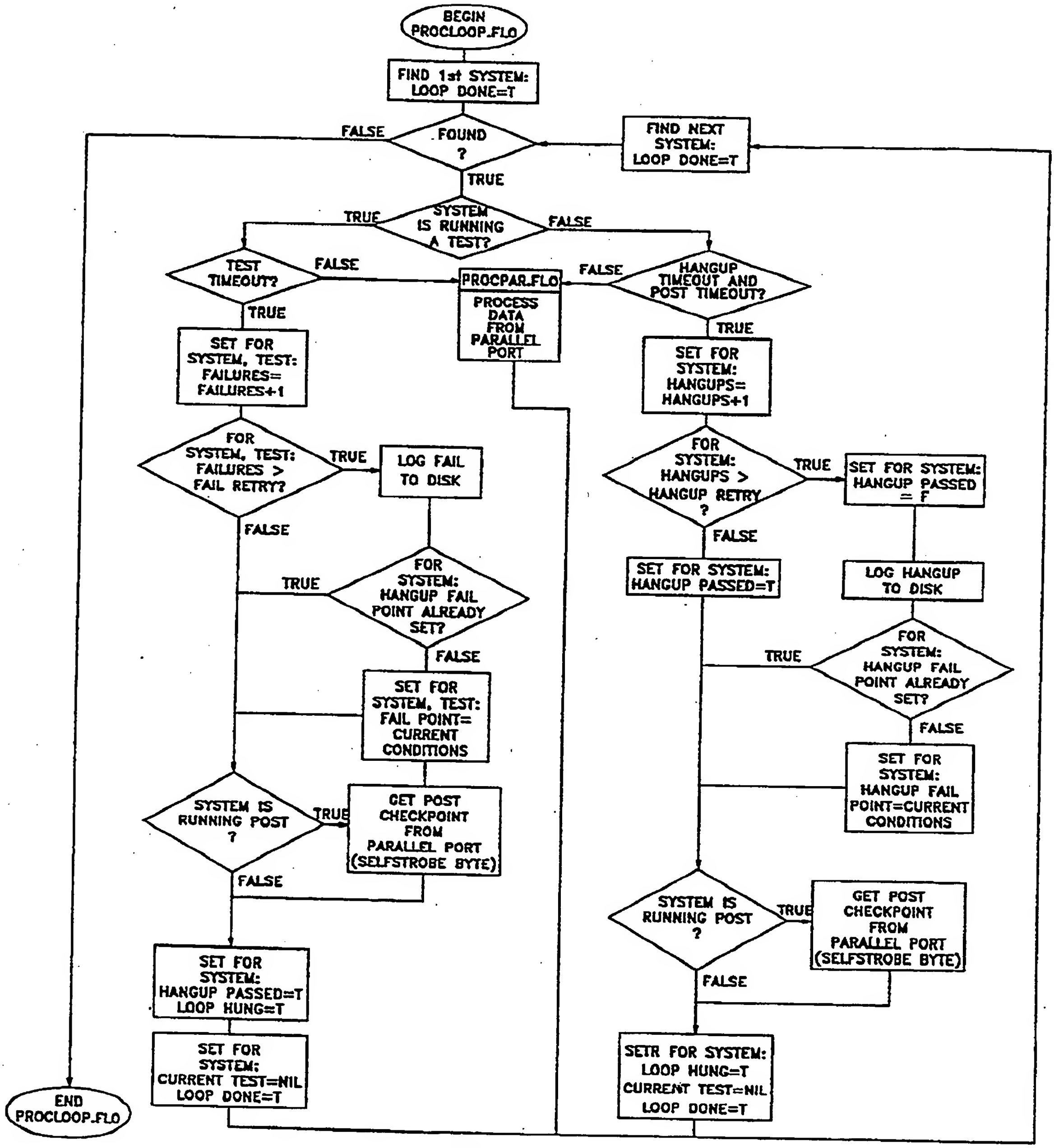
BEGIN PROCLOOP.FLO
FIND 1st SYSTEM: LOOP DONE=T
FOUND ?
FALSE
FIND NEXT SYSTEM: LOOP DONE=T
TRUE
SYSTEM IS RUNNING A TEST?
TRUE
FALSE
TEST TIMEOUT?
FALSE
TRUE
PROCPAR.FLO
PROCESS DATA FROM PARALLEL PORT
FALSE
HANGUP TIMEOUT AND POST TIMEOUT?
TRUE
SET FOR SYSTEM, TEST: FAILURES= FAILURES+1
SET FOR SYSTEM: HANGUPS= HANGUPS+1
FOR SYSTEM, TEST: FAILURES > FAIL RETRY?
TRUE
LOG FAIL TO DISK
FOR SYSTEM: HANGUPS > HANGUP RETRY ?
TRUE
SET FOR SYSTEM: HANGUP PASSED = F
FALSE
FOR SYSTEM: HANGUP FAIL POINT ALREADY SET?
TRUE
FALSE
SET FOR SYSTEM: HANGUP PASSED=T
LOG HANGUP TO DISK
FOR SYSTEM: HANGUP FAIL POINT ALREADY SET?
TRUE
FALSE
SET FOR SYSTEM, TEST: FAIL POINT= CURRENT CONDITIONS
SET FOR SYSTEM: HANGUP FAIL POINT=CURRENT CONDITIONS
SYSTEM IS RUNNING POST ?
TRUE
GET POST CHECKPOINT FROM PARALLEL PORT (SELFSTROBE BYTE)
SYSTEM IS RUNNING POST ?
TRUE
GET POST CHECKPOINT FROM PARALLEL PORT (SELFSTROBE BYTE)
FALSE
FALSE
SET FOR SYSTEM: HANGUP PASSED=T LOOP HUNG=T
SETR FOR SYSTEM: LOOP HUNG=T CURRENT TEST=NIL LOOP DONE=T
SET FOR SYSTEM: CURRENT TEST=NIL LOOP DONE=T
END PROCLOOP.FLO

FIG. 4

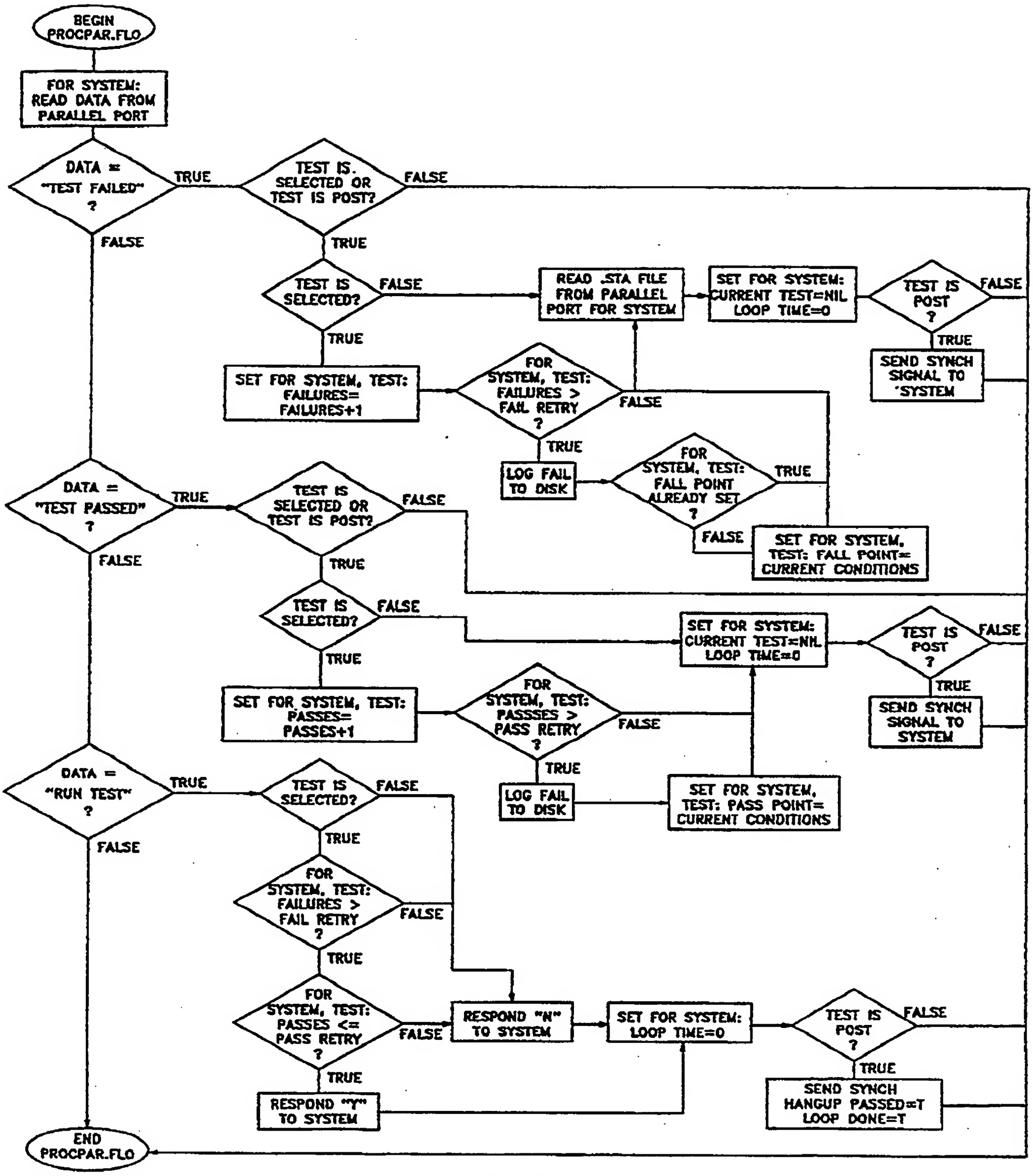
BEGIN PROCPAR.FLO
FOR SYSTEM: READ DATA FROM PARALLEL PORT
DATA = "TEST FAILED" ?
TRUE
FALSE
TEST IS SELECTED OR TEST IS POST?
FALSE
TRUE
TEST IS SELECTED?
FALSE
TRUE
READ .STA FILE FROM PARALLEL PORT FOR SYSTEM
SET FOR SYSTEM: CURRENT TEST=NIL LOOP TIME=0
TEST IS POST ?
FALSE
TRUE
SEND SYNCH SIGNAL TO SYSTEM
SET FOR SYSTEM, TEST: FAILURES= FAILURES+1
FOR SYSTEM, TEST: FAILURES > FAIL RETRY ?
FALSE
TRUE
LOG FAIL TO DISK
FOR SYSTEM, TEST: FALL POINT ALREADY SET ?
TRUE
FALSE
SET FOR SYSTEM, TEST: FALL POINT= CURRENT CONDITIONS
DATA = "TEST PASSED" ?
TRUE
FALSE
TEST IS SELECTED OR TEST IS POST?
FALSE
TRUE
TEST IS SELECTED?
FALSE
TRUE
SET FOR SYSTEM: CURRENT TEST=NIL LOOP TIME=0
TEST IS POST ?
FALSE
TRUE
SEND SYNCH SIGNAL TO SYSTEM
SET FOR SYSTEM, TEST: PASSES= PASSES+1
FOR SYSTEM, TEST: PASSSES > PASS RETRY ?
FALSE
TRUE
LOG FAIL TO DISK
SET FOR SYSTEM, TEST: PASS POINT= CURRENT CONDITIONS
DATA = "RUN TEST" ?
TRUE
FALSE
TEST IS SELECTED?
FALSE
TRUE
FOR SYSTEM, TEST: FAILURES > FAIL RETRY ?
FALSE
TRUE
FOR SYSTEM, TEST: PASSES <= PASS RETRY ?
FALSE
TRUE
RESPOND "N" TO SYSTEM
SET FOR SYSTEM: LOOP TIME=0
TEST IS POST ?
FALSE
TRUE
SEND SYNCH HANGUP PASSED=T LOOP DONE=T
RESPOND "Y" TO SYSTEM
END PROCPAR.FLO

FIG. 5

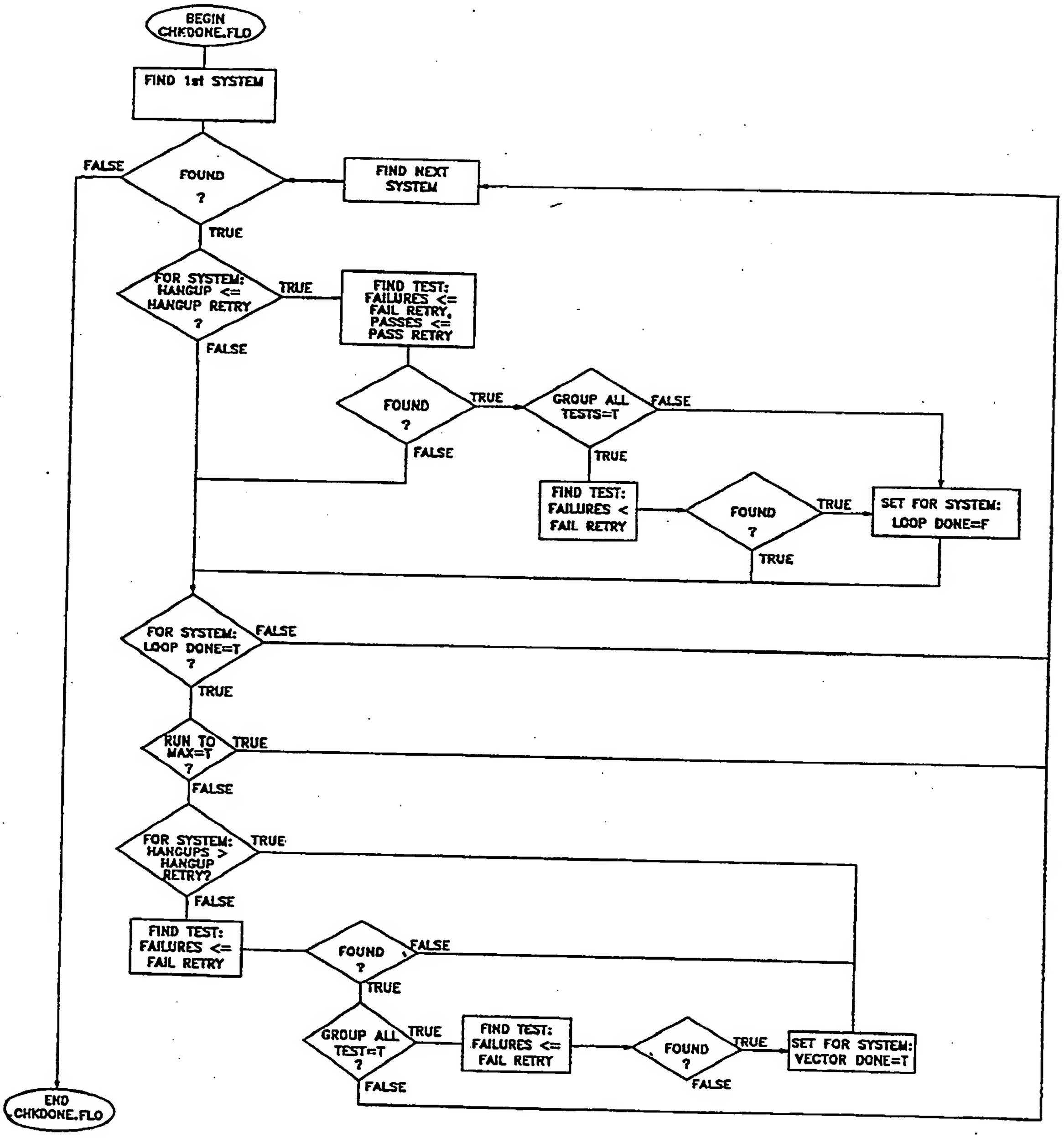
BEGIN CHKDONE.FLO
FIND 1st SYSTEM
FOUND ?
FALSE
TRUE
FIND NEXT SYSTEM
FOR SYSTEM: HANGUP <= HANGUP RETRY ?
TRUE
FALSE
FIND TEST: FAILURES <= FAIL RETRY, PASSES <= PASS RETRY
FOUND ?
TRUE
FALSE
GROUP ALL TESTS=T
FALSE
TRUE
FIND TEST: FAILURES < FAIL RETRY
FOUND ?
TRUE
TRUE
SET FOR SYSTEM: LOOP DONE=F
FOR SYSTEM: LOOP DONE=T ?
FALSE
TRUE
RUN TO MAX=T ?
TRUE
FALSE
FOR SYSTEM: HANGUPS > HANGUP RETRY?
TRUE
FALSE
FIND TEST: FAILURES <= FAIL RETRY
FOUND ?
FALSE
TRUE
GROUP ALL TEST=T ?
TRUE
FALSE
FIND TEST: FAILURES <= FAIL RETRY
FOUND ?
TRUE
FALSE
SET FOR SYSTEM: VECTOR DONE=T
END CHKDONE.FLO

FIG. 6

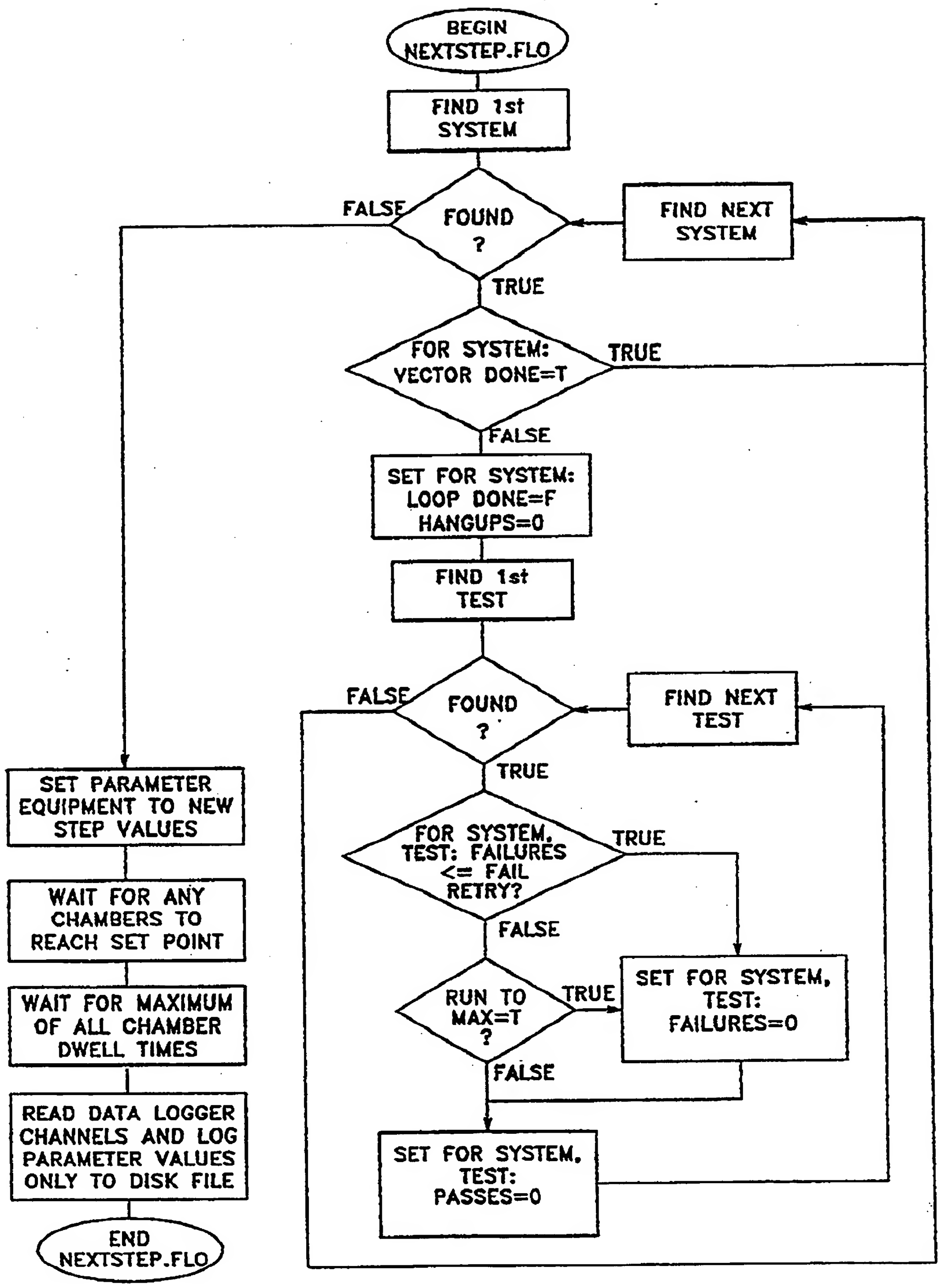
BEGIN NEXTSTEP.FLO
FIND 1st SYSTEM
FOUND ?
FALSE
TRUE
FIND NEXT SYSTEM
FOR SYSTEM: VECTOR DONE=T
TRUE
FALSE
SET FOR SYSTEM: LOOP DONE=F HANGUPS=0
FIND 1st TEST
FOUND ?
FALSE
TRUE
FIND NEXT TEST
FOR SYSTEM, TEST: FAILURES <= FAIL RETRY?
TRUE
FALSE
RUN TO MAX=T ?
TRUE
SET FOR SYSTEM, TEST: FAILURES=0
FALSE
SET FOR SYSTEM, TEST: PASSES=0
SET PARAMETER EQUIPMENT TO NEW STEP VALUES
WAIT FOR ANY CHAMBERS TO REACH SET POINT
WAIT FOR MAXIMUM OF ALL CHAMBER DWELL TIMES
READ DATA LOGGER CHANNELS AND LOG PARAMETER VALUES ONLY TO DISK FILE
END NEXTSTEP.FLO

FIG. 7

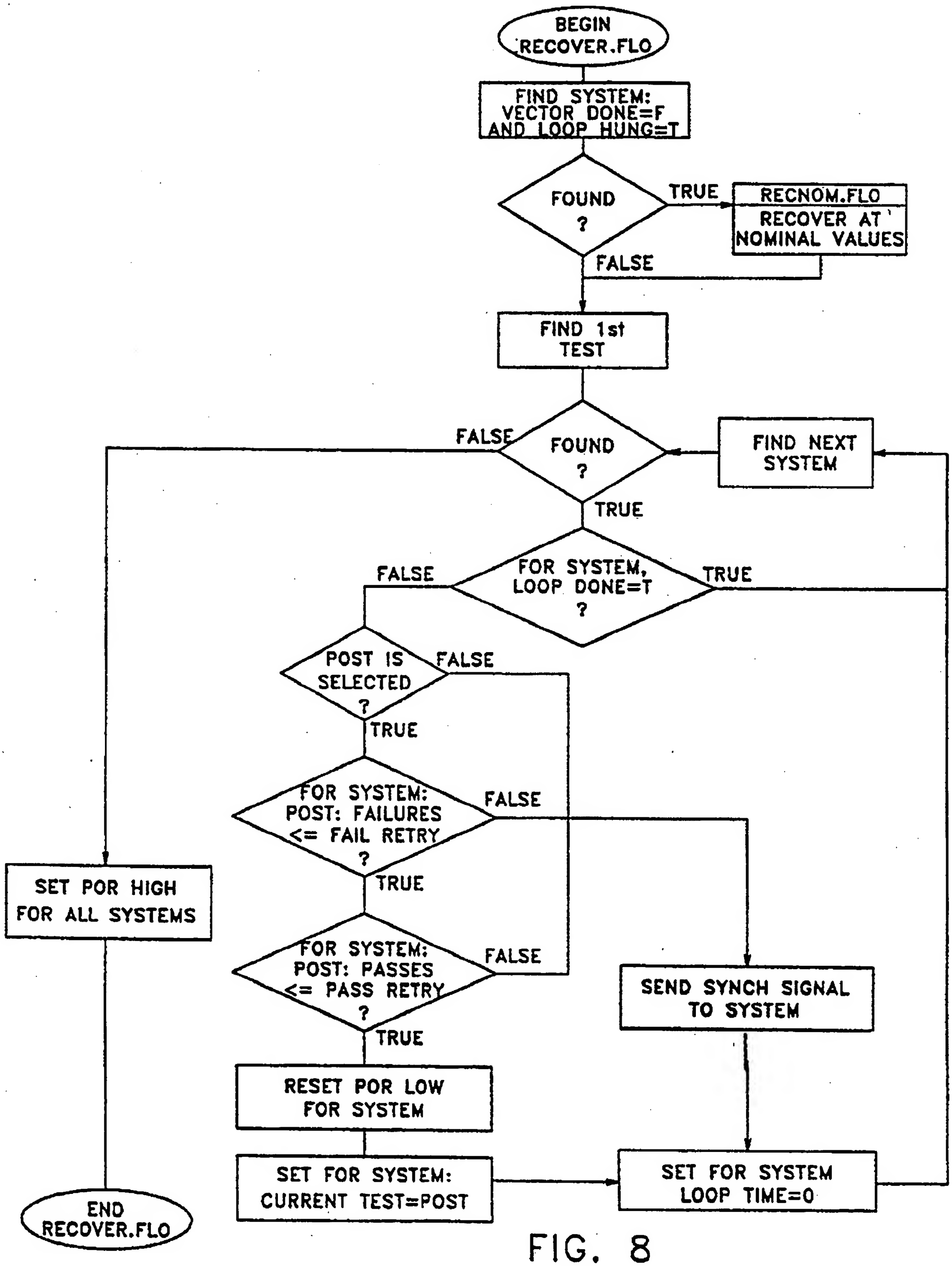
BEGIN RECOVER.FLO
FIND SYSTEM: VECTOR DONE=F AND LOOP HUNG=T
FOUND ?
TRUE
RECNOM.FLO
RECOVER AT NOMINAL VALUES
FALSE
FIND 1st TEST
FOUND ?
FALSE
FIND NEXT SYSTEM
TRUE
FOR SYSTEM, LOOP DONE=T ?
FALSE
TRUE
POST IS SELECTED ?
FALSE
TRUE
FOR SYSTEM: POST: FAILURES <= FAIL RETRY ?
FALSE
TRUE
FOR SYSTEM: POST: PASSES <= PASS RETRY ?
FALSE
TRUE
SET POR HIGH FOR ALL SYSTEMS
RESET POR LOW FOR SYSTEM
SET FOR SYSTEM: CURRENT TEST=POST
SEND SYNCH SIGNAL TO SYSTEM
SET FOR SYSTEM LOOP TIME=0
END RECOVER.FLO

FIG. 8

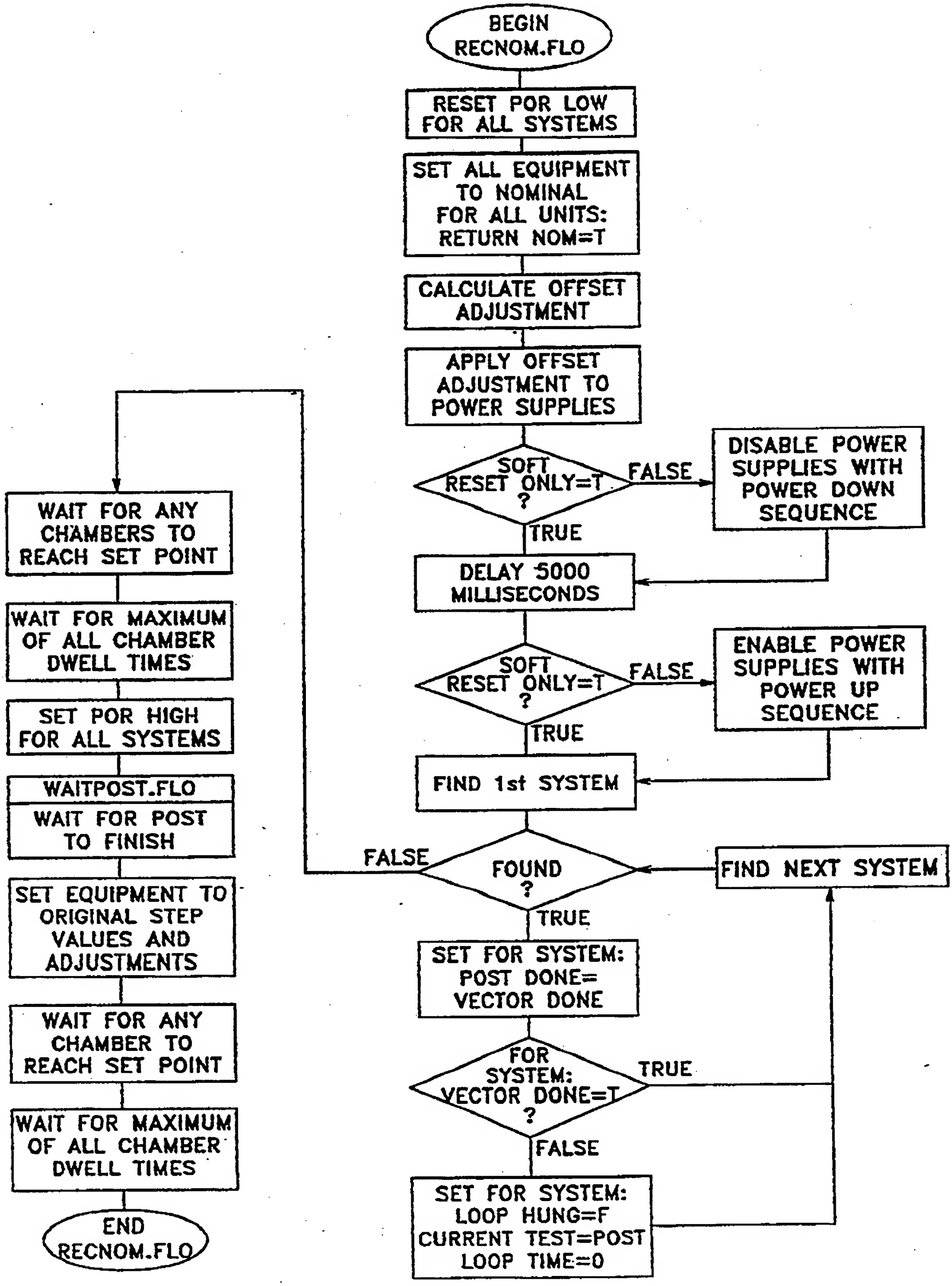
BEGIN RECNOM.FLO
RESET POR LOW FOR ALL SYSTEMS
SET ALL EQUIPMENT TO NOMINAL FOR ALL UNITS: RETURN NOM=T
CALCULATE OFFSET ADJUSTMENT
APPLY OFFSET ADJUSTMENT TO POWER SUPPLIES
SOFT RESET ONLY=T ?
FALSE
DISABLE POWER SUPPLIES WITH POWER DOWN SEQUENCE
TRUE
DELAY 5000 MILLISECONDS
SOFT RESET ONLY=T ?
FALSE
ENABLE POWER SUPPLIES WITH POWER UP SEQUENCE
TRUE
FIND 1st SYSTEM
FOUND ?
FALSE
FIND NEXT SYSTEM
TRUE
SET FOR SYSTEM: POST DONE= VECTOR DONE
FOR SYSTEM: VECTOR DONE=T ?
TRUE
FALSE
SET FOR SYSTEM: LOOP HUNG=F CURRENT TEST=POST LOOP TIME=0
WAIT FOR ANY CHAMBERS TO REACH SET POINT
WAIT FOR MAXIMUM OF ALL CHAMBER DWELL TIMES
SET POR HIGH FOR ALL SYSTEMS
WAITPOST.FLO
WAIT FOR POST TO FINISH
SET EQUIPMENT TO ORIGINAL STEP VALUES AND ADJUSTMENTS
WAIT FOR ANY CHAMBER TO REACH SET POINT
WAIT FOR MAXIMUM OF ALL CHAMBER DWELL TIMES
END RECNOM.FLO

FIG. 9

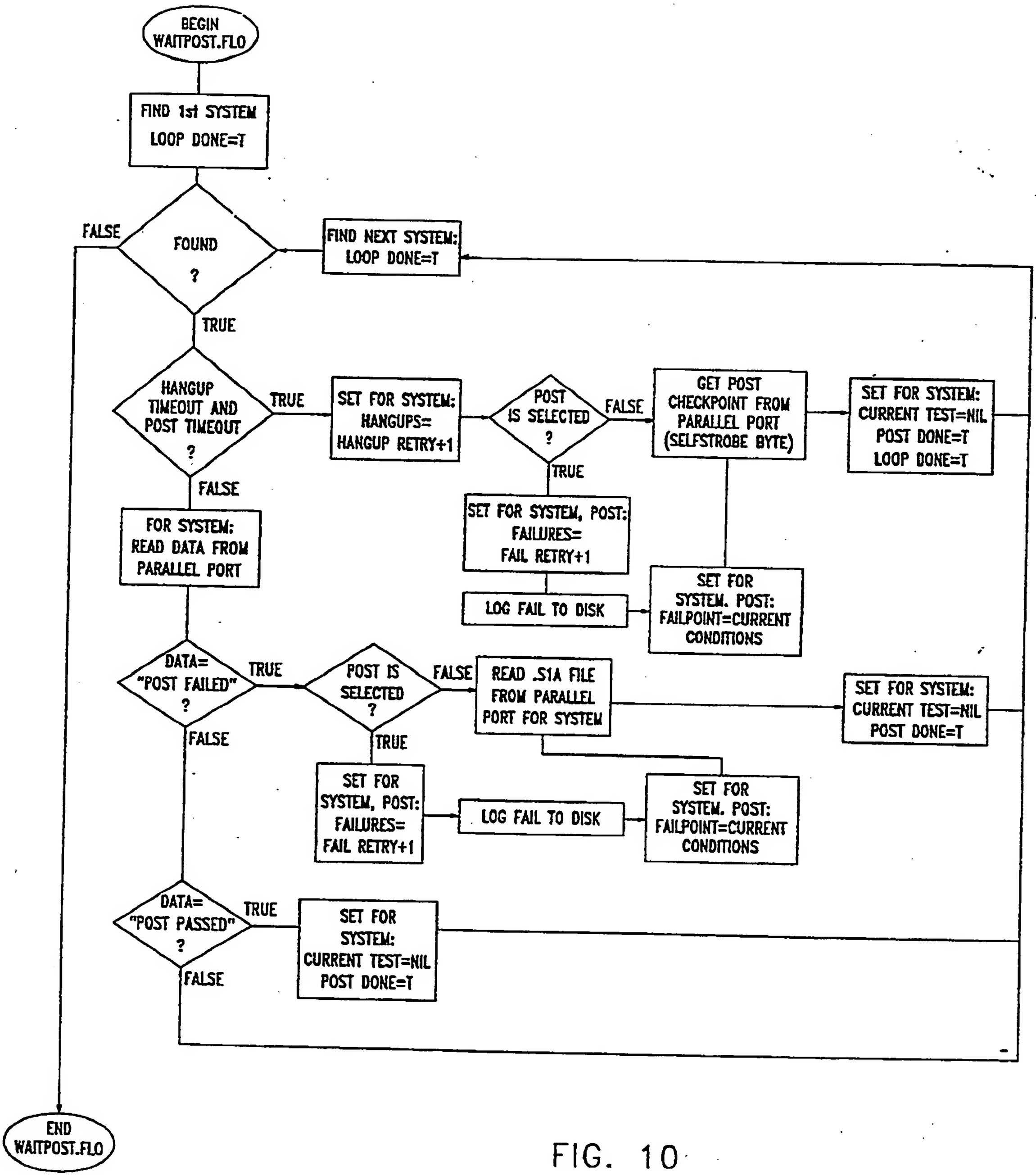
BEGIN WAITPOST.FLO
FIND 1st SYSTEM LOOP DONE=T
FOUND ?
FALSE
TRUE
FIND NEXT SYSTEM: LOOP DONE=T
HANGUP TIMEOUT AND POST TIMEOUT ?
TRUE
SET FOR SYSTEM: HANGUPS= HANGUP RETRY+1
POST IS SELECTED ?
FALSE
GET POST CHECKPOINT FROM PARALLEL PORT (SELFSTROBE BYTE)
SET FOR SYSTEM: CURRENT TEST=NIL POST DONE=T LOOP DONE=T
TRUE
SET FOR SYSTEM, POST: FAILURES= FAIL RETRY+1
LOG FAIL TO DISK
SET FOR SYSTEM. POST: FAILPOINT=CURRENT CONDITIONS
FALSE
FOR SYSTEM: READ DATA FROM PARALLEL PORT
DATA= "POST FAILED" ?
TRUE
POST IS SELECTED ?
FALSE
READ .S1A FILE FROM PARALLEL PORT FOR SYSTEM
SET FOR SYSTEM: CURRENT TEST=NIL POST DONE=T
TRUE
SET FOR SYSTEM, POST: FAILURES= FAIL RETRY+1
LOG FAIL TO DISK
SET FOR SYSTEM. POST: FAILPOINT=CURRENT CONDITIONS
FALSE
DATA= "POST PASSED" ?
TRUE
SET FOR SYSTEM: CURRENT TEST=NIL POST DONE=T
FALSE
END WAITPOST.FLO

FIG. 10